In [154]:

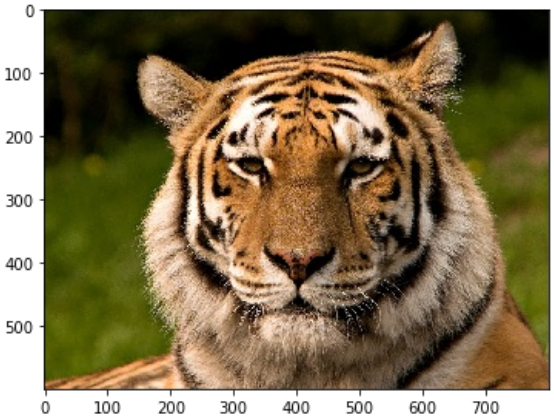
```
from skimage.io import imread, imshow
from skimage import img_as_float
import math
%matplotlib inline
```

In [155]:

```
img = imread('tiger-color.png')
entropy = {}
imshow(img)
```

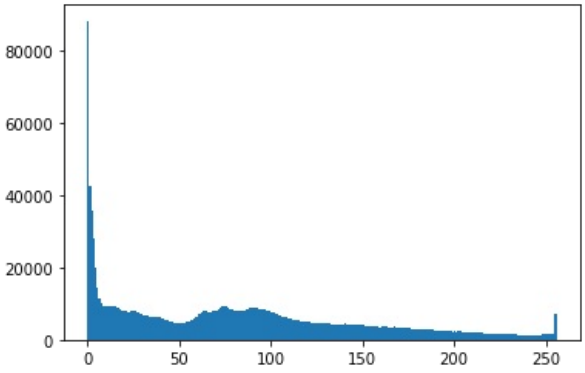
Out[155]:

```
<matplotlib.image.AxesImage at 0x197e2430>
```



In [156]:

```
from matplotlib.pyplot import hist
values1, bin_edges1, patches1 = hist(img.ravel(), bins=range(257))
```



In [157]:

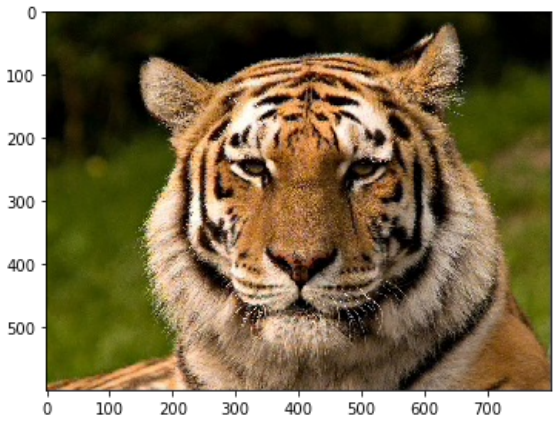
```
H1 = []
for i in range(256):
    pi = values1[i]/(img.shape[0]*img.shape[1])
    if pi != 0:
        H1.append(-pi * math.log(pi))
    else:
        H1.append(0)
entropy.update({"just picture": sum(H1)})
```

In [158]:

```
# lab 1.2
img1 = imread('decimial.png')
imshow(img1)
```

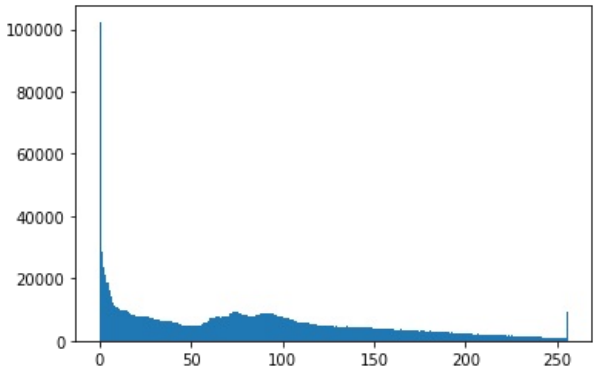
Out[158]:

```
<matplotlib.image.AxesImage at 0x155aa6b8>
```



In [159]:

```
from matplotlib.pyplot import hist
values1, bin_edges1, patches1 = hist(img1.ravel(), bins=range(257))
```



In [160]:

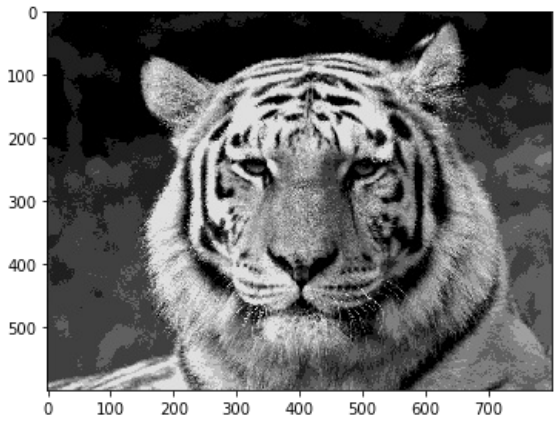
```
H2 = []
for i in range(256):
    pi = values1[i]/(img1.shape[0]*img1.shape[1])
    if pi != 0:
        H2.append(-pi * math.log(pi))
    else:
        H2.append(0)
entropy.update({"Децимация": sum(H2)})
```

In [161]:

```
# lab 1.1
img2 = imread('lab11.png')
imshow(img2)
```

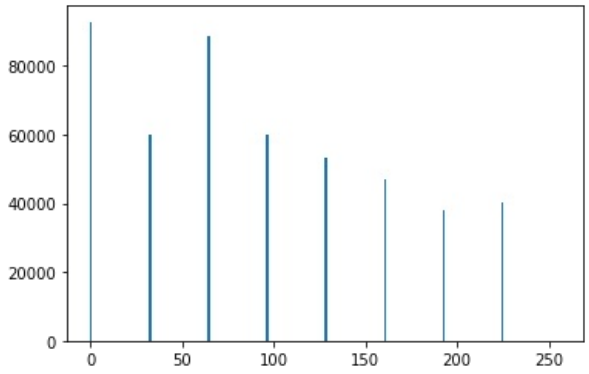
Out[161]:

```
<matplotlib.image.AxesImage at 0x4930b20>
```



In [162]:

```
from matplotlib.pyplot import hist
values1, bin_edges1, patches1 = hist(img2.ravel(), bins=range(257))
```



In [163]:

```
H3 = []
for i in range(256):
    pi = values1[i]/(img2.shape[0]*img2.shape[1])
    if pi != 0:
        H3.append(-pi * math.log(pi))
    else:
        H3.append(0)
entropy.update({"Пототечные методы": sum(H3)})
```

In [164]:

```
entropy
```

Out[164]:

```
{'just picture': 12.357977027303733,
 'Децимация': 12.355716034750934,
 'Пототечные методы': 2.0296896618600657}
```

In [ ]:

In [172]:

```
def MSE(img, img1):
    if len(img.shape) > 2:
        img = img[:,:,0]
    if len(img1.shape) >2:
        img1 = img1[:,:,0]
    mse = 0
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            mse += math.pow((img[i,j] - img1[i,j]), 2)
    mse = mse / (img.shape[1] * img.shape[0])
    return mse
```

In [173]:

```
print(MSE(img, img1))
print(MSE(img, img2))
```

```
<ipython-input-172-05f0f5d578f2>:9: RuntimeWarning: overflow encountered in ubyte_scalars
  mse += math.pow((img[i,j] - img1[i,j]), 2)

24176.959070833334
323.1694708333333
```